Weekly Report

1. 总览

(1) 实现了神经网络
(2) 对多种可能的情况作了测试
(3) 阅读相关文献，提出进一步的构想

2. 工作概述

2.1 上周工作回顾

上周我提出了如图 1 所示的 pipeline，而本周工作的重点为解决 MMA (map matching algorithm)的实现算法。

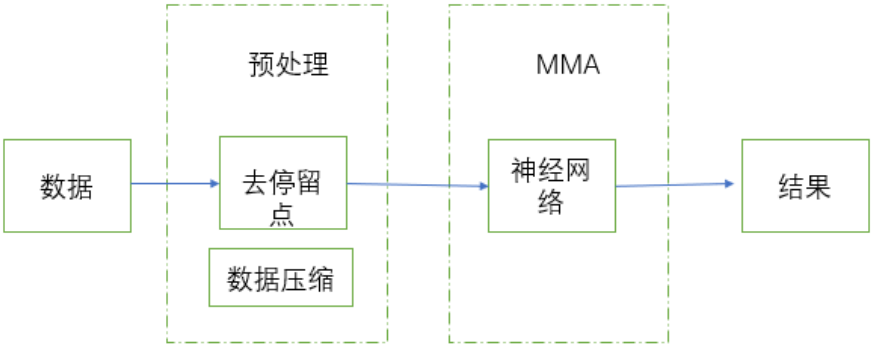


图 1

用神经网络实现的 MMA，思路是特征工程，其基本过程可以分为如图 2 所示的三个过程: (1) 输入数据提取，由于地图数据非常大，不易将所有数据输入算法中，因此先选择数据输入的范围及类型；(2) 神经网络，所用到的神经网络的结构；(3) 后处理，对生成的结果进行后处理以生成最终结果。

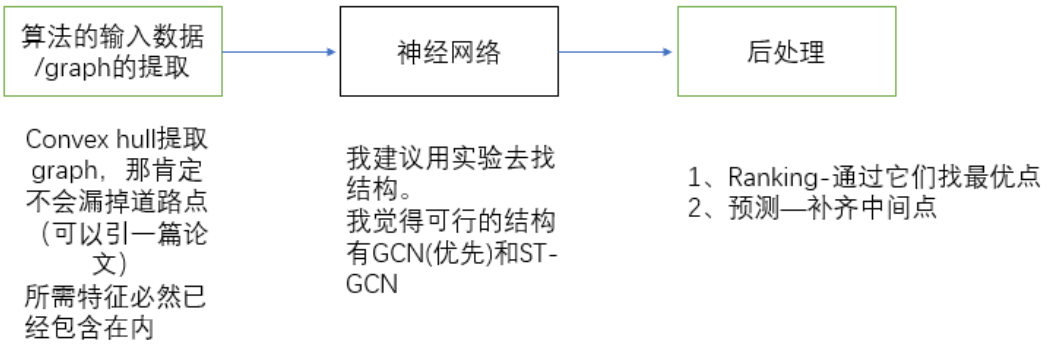


图 2

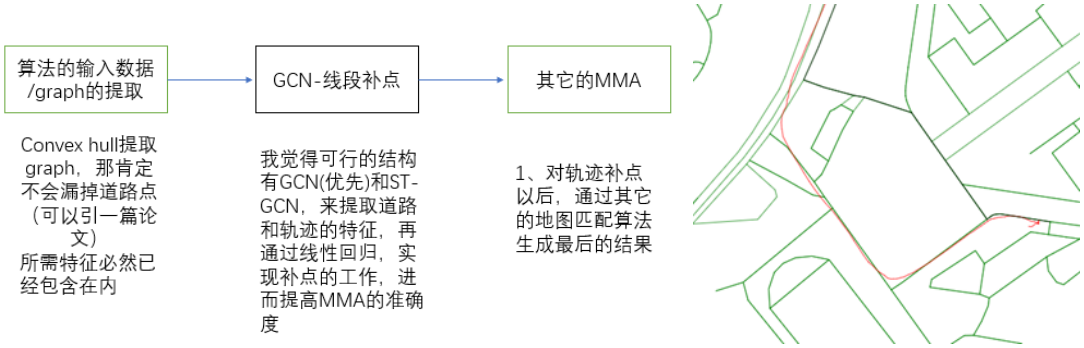
输入数据的类型选择上，我开始选用的是 image，后来实验证明 CNN 难以捕捉到线段的特征。因此考虑采用 graph 作为输入数据。从输入数据的范围考虑，由于汽车有最大行驶速度，那么两 GPS 点的汽车可能的路径就只能在某一有限的范围内，由之前的文献表明，这一范围大致是个椭圆，类似于图形学中的 convex hull 概念。在 section 2 的后续部分将会阐述后面两个过程的解决方案。

2.2 提出的两种实现方法

针对 MMA 的实现，我提出了 2 种可能的构想。

2.2.1 最初的方案

最初的方法如图 3 (1)所示，用 Graph Convolutional Neural Network 提取网络的特征，再通过线性回归得到缺失点的位置。补缺轨迹后，再用其它 MMA，利用补全后的轨迹生成最终的结果。如图 3 (2)所示，黑线为因缺失点导致了错误的结果。



(1) (2)

图 3

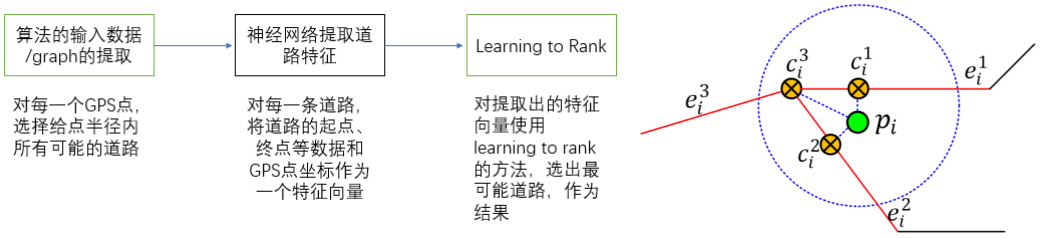
综合分析以后，我们认为，这种方法可以做。这条研究路线的缺点在于：（1）它实质上是一个轨迹预测问题，且是一种轨迹预测问题的简化版。现有的大多数轨迹预测方法，可以较 instinctively 移植到该问题上。这种思路的研究较少，也可以做。（2）它所需要的数据量会很大，需要考虑如何构造数据，实现比较困难

2.2.2 更简易的方案

在阅读文献思考解决方案一的过程中，我想到了方案二，如图 4 （1）所示。通过阅读文献，有人用 fuzzy theory 来生成最终道路的选择可能性大小。那么神经网络一样能做这个工作。阅读文献，发现大部分问题用于判断某一 GPS 点属于哪条道路用到的信息可以分为以下两类：（1）局部特征 （2）全局特征。因此我打算对这两种特征分别采用两种方法去处理。由于很多文献，只用局部信息，就可以获得较好结果，故暂时先只考虑局部信息的情况。

初步的打算是，首先对每个 GPS 点提取出在某一半径内的道路，作为 candidate。查阅相关文献发现，很多人工设计的特征是用低阶函数对 candidate 的坐标及附近道路的坐标变换而来。神经网络适于逼近函数，因此对每个 candidate，用它的坐标和 GPS 点的坐标作为一个特征向量 X^i = {Lon1,lat1,lon2,lat2,…}。再用神经网络对这些特征向量作变换，$V^i = F(X^i)$，得新的特征 $V^i = \{v_1, V_2, V_3, \cdots\}$ 。当然，这一过程也能够使用图神经网络作为替代。目前神经网络最基本的两个任务是分类和回归。如何实现 rank 是一个问题，因此我觉得借鉴相关文献中的研究资料，找到了 list-wise learning to rank 方法。学到一个 ranking 函数。通过这个 ranking 函数，$rank(V^i)$，选择 top-one 作为 GPS 点所对应的道路，再利用 graph 的最短路径搜索算法补全两道路之间的路线，即可得到 Map Matching 的结果。学习方法为梯度下降法。

验证该方案可以保证同时验证方案一的可行性，实现也更简单。因此我决定优先验证该方法。后续可能的工作是，利用 LSTM 等方法提取全局的特征进一步提高匹配的准确度。实际上 HMM 的优势正是利用了全局特征，才得到了比 fuzzy theory 方法更高的匹配准确度。因此初步预计方法我的方法会比 HMM 略差，但有明显的效果。



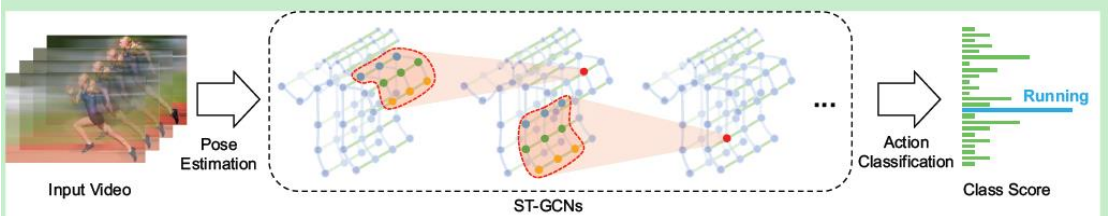
(1) (2)

图 4

3. 论文阅读

《Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition》

The article proposed a spatial temporal graph convolutional network method for skeleton based action recognition. But the method doesn't consider using edge weight, but consider using edge connectivity. For our task, we may can use this network structure.



《A High Accuracy Fuzzy Logic Based Map Matching Algorithm for Road Transport》

An article which use fuzzy logic and multi-source fusion method for map matching. Maybe we could use its way that utilize fuzzy logic to help map match to inspirit conceiving our method. It uses different types of feature to decide the probabilities of candidate links.

- *If (v is high) and (HE is small) then (L1 is average) (3)*
- *If (v is high) and (HE is large) then (L1 is low) (1)*
- *If (HDOP is good) and (PD is short) then (L1 is average) (1)*
- *If (HDOP is good) and (PD is long) then (L1 is low) (1)*
- *If (HE is small) and (PD is short) then (L1 is high) (1)*
- *If (HE is large) and (PD is long) then (L1 is low) (1)*

《The Path Inference Filter: Model-Based Low-Latency Map Matching of Probe Vehicle Data》

This article proposed a general framework, which is the same as HMM-based MMA. Due to conditional random field rather than HMM, this model using $P(o|x)$ instead of $P(x|o)$ in which x is real position and o is observed position.

$$\phi(\tau^{1:t}|g^{1:t}) = \omega(g^1|x^1) \prod_{t'=1}^{t-1} \underline{\delta}\left(x^{t'}, p^{t'}\right) \eta\left(p^{t'}\right) \cdot \bar{\delta}\left(p^{t'}, x^{t'+1}\right) \omega\left(g^{t'+1}|x^{t'+1}\right). \quad (7)$$

《AntMapper: An Ant Colony-Based Map Matching Approach for Trajectory-Based Applications》

This article proposed a method which use ant colony algorithm instead of Viterbi algorithm to solve HMM (hidden Marko Model) optimal problem. This boring an advantage that we can use global criteria to constrain the final result of a map matching algorithm.

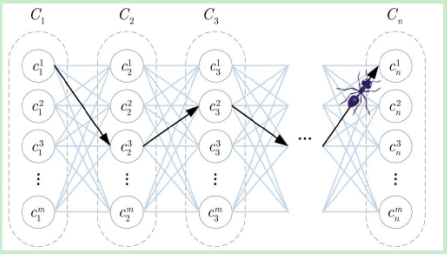


Fig. 5. Illustration of an ant constructing a path on the information graph. The candidate solution obtained is $c_1^1 \to c_2^3 \to c_3^2 \to \cdots \to c_n^1$.

stored in an information graph (as shown in Fig. 5, each vertex is a candidate matched points), and the global fitness function is formulated. Note that we should not evaluate the global fitness of a path in this module since we do not build a candidate path until the third module starts. Finally, in the solution finding module, a colony of ants is dispatched

Algorithm 1 AntMapper

```
/* Initialization */
Set the pheromone on all links of the information graph to an equal initial value;
/* Main Loop */
while Termination Rule is not met do
    Place each ant at a random starting point in the candidate set of the first GPS point (i = 1);
    for i = 2 to n do
        Perform State Transition Rule to move each ant to the next point;
        Perform Local Update Rule to update the pheromone on the link which is just passed by each ant;
    end for
    Evaluate the fitness values of all ants according to (6);
    Identify the best-so-far path P_bsf;
    Perform Global Update Rule to update the pheromone on path P_bsf;
end while
return Path P_bsf.
```

《Learning to Rank using Gradient Descent》

It's a pair-wise method which uses two layers neural network to rank data.

《Learning to Rank From Pairwise Approach to Listwise Approach》

It's a list-wise method which uses cross entropy to rank data. It provide a concluding, and a formulation of loss.

Again, let us take document retrieval as example. We denote the ranking function based on the Neural Network model ω as f_ω. Given a feature vector $x_j^{(i)}$, $f_\omega(x_j^{(i)})$ assigns a score to it. We define ϕ in Definition 1 as an exponential function, which is translation invariant as shown in Theorem 5. We then rewrite the top k probability in Theorem 8 as

$$P_s(\mathcal{G}_k(j_1, j_2, ..., j_k)) = \prod_{t=1}^{k} \frac{\exp(s_{j_t})}{\sum_{l=t}^{n^{(i)}} \exp(s_{j_l})},$$

Given query $q^{(i)}$, the ranking function f_ω can generate a score list $z^{(i)}(f_\omega) = \left(f_\omega(x_1^{(i)}), f_\omega(x_2^{(i)}), \cdots, f_\omega(x_{n^{(i)}}^{(i)})\right)$. Then the top k probability of documents $(d_{j_1}^{(i)}, d_{j_2}^{(i)}, ..., d_{j_k}^{(i)})$ is calculated as

$$P_{z^{(i)}(f_\omega)}(\mathcal{G}_k(j_1, j_2, ..., j_k)) = \prod_{t=1}^{k} \frac{\exp(f_\omega(x_{j_t}^{(i)}))}{\sum_{l=t}^{n^{(i)}} \exp(f_\omega(x_{j_l}^{(i)}))},$$

With Cross Entropy as metric, the loss for query $q^{(i)}$ becomes

每一种排列为一个类别，即所有排列为一个概率分布，最后用交叉熵loss

$$L(y^{(i)}, z^{(i)}(f_\omega)) = -\sum_{\forall g \in \mathcal{G}_k} P_{y^{(i)}}(g) \log(P_{z^{(i)}(f_\omega)}(g)) \quad (4)$$

The gradient of $L(y^{(i)}, z^{(i)}(f_\omega))$ with respect to parameter ω can be calculated as follows

$$\Delta\omega = \frac{\partial L(y^{(i)}, z^{(i)}(f_\omega))}{\partial \omega} = -\sum_{\forall g \in \mathcal{G}_k} \frac{\partial P_{z^{(i)}(f_\omega)}(g)}{\partial \omega} \frac{P_{y^{(i)}}(g)}{P_{z^{(i)}(f_\omega)}(g)} \quad (5)$$

4. 时间安排

Date	Tasks	Duration
Mon.	Reading and Programming	10:30-22:00
Tues. to Sat.	Reading and Programming	9:00-22:00
Sun.	Reading and Programming	9:30-22:00

Work Time：above 50 hours